DRA: Distributionally Robust Adversarial Training

Nian Si
MS&E Stanford
niansi@stanford.edu

Fan Zhang
MS&E Stanford
fzh@stanford.edu

Teng Zhang
MS&E Stanford
tengz@stanford.edu

Abstract

In this paper, we generalize the work of [19] based on distributional robust optimization framework to improve the model's robustness against adversarial examples at scale. We develop the defense framework of multi-cost ensemble and adaptive cost learning. We test our method by training a 13 layer neural net on the CIFAR-10 dataset against various white-box attacks. The result shows that the proposed defense achieves the state-of-art performance in ℓ_∞ based attacks and outperforms other methods in ℓ_2 based attacks.

1. Introduction

Neural network is a powerful model that has excellent explainary power in the domain of vision classifications and speech recognitions, etc. However, it is well known that neural network is vulnerable to adversarial examples: even imperceptible perturbations to legistimate data may result in misclassifications. In the context such as automatic driving where mistakes are expensive, the robustness of neural network to adversarial examples received increasing attention from machine learning community.

To remedy the vulnerability to adversarial examples, many methodologies are proposed to generate and defend against the adversarial attacks. When the model structure is available, researchers proposed many algorithms to generate adversarial examples, including box-constrained L-BFGS [20], fast gradient sign method [8], iterative gradient method [11], momentum gradient method [5] and DeepFool [15].

All the aforementioned attacks are heuristic, and consequently there is no theoretical performance guarantee on the related adversarial training procedure. Recently, Sinha et.al. developed an adversarial training procedure from the perspective of distributionally robust optimization [19]. They also conducted insightful statistical analysis that guarantees the performence when input is perturbed by adversary.

In this paper, we are going to explore the method proposed by [19] in depth, and improve their algorithm from the following aspects:

1. We show our distributionally robust adversarial training framework encompasses many other popular heuristic methods used in adversarial training.
2. We generalize the method proposed by [19] and test our method in larger data sets (CIFAR10).
3. We apply attack ensemble techniques to robustify our defend method.

The input to our neural network is a picture in CIFAR10 dataset. We use a convolutional neural network with 10 CNN layers to perform the distributionally robust adversarial training. During the training process, we alternate between updates of adversarial example with updates of model parameter. Eventually, our neural network after training is able to classify pictures under various attacks. In addition, the training scheme will also produce an adversarial example that almost indistinguishable from the input picture.

The rest of this paper is organized as follows. In Section 2, we review some existing attack and defense mechanisms in literatures. In Section 3, we develop our distributionally robust adversarial training procedure. In section 4, we present some numerical results when our adversarial training procedure is applied to CIFAR10 dataset. In Section 5, we give some intuitive anlysis. The proof of a technical theorem is deferred to Appendix.

2. Background on Attacks and Defenses

To be more precise about the notion of "attack" and "defense" in the existing adversarial training literature, we phrase them primarily in the following two questions:

1. How can we generate powerful adversarial examples with minimum perturbation, even imperceptible to human eyes?
2. How can we train a model that is robust to all possible adversarial examples?

We will discuss existing works on both attack and defense sides.

2.1. Attacks

Attack methodologies basically fall in two categories: optimization-based attacks and gradient-based attacks.

2.1.1 Optimization-based Attacks

DeepFool method [16] tried to solve

$$\min_r \|r\|_2 : s.t.\ C(x+r) \neq f(x),$$

where $C(x)$ is the classifier.

And similarly, L-BFGS [20]

$$\min_r \|r\|_2 : s.t.\ C(x+r) = l, (x+r) \in [0,1]^m,$$

where l is the targeted label. To make it tractable, they minimize the penalty function alternatively,

$$\min_r c|r| + l_l(\theta; x+r),$$

where $l_l(\theta; x+r)$ is the loss with the target label l. [2] use a more generalized function $f(\theta; x)$ and the formulate the problem as

$$\min_r D(x, x+r) + f(\theta; x+r),$$

where D is a regularizer. Finally, Elastic-Net Attacks [3] used a Elastic-Net regularizer:

$$\min_r f(\theta; x+r) + \lambda_1 \|r\|_1 + \lambda_2 \|r\|_2^2 .$$

2.1.2 Gradient-based Attacks

The first work is fast gradient sign method (FGM) [9]. FGM is an attack for an ℓ_∞-bounded adversary that computes an adversarial example as

$$\hat{x} = x + \epsilon \operatorname{sgn}(\nabla_x l(\theta, x, y)). \tag{1}$$

And a powerful attack is its iteration variation (BIM) [14, 11],

$$\hat{x}^{t+1} = \Pi_{x+S}\left(\hat{x}^t + \alpha \operatorname{sgn}(\nabla_x l(\theta, \hat{x}^t, y))\right), \hat{x}^0 = x. \tag{2}$$

where S is a small ball around x. This method can be viewed as finding a local maximum of the loss function $l(\cdot)$ around the original image x. Recently, Momentum Iteration Method (MIM) is proposed in [5] to accelerate the process of finding maximum,

$$\begin{cases} g^{t+1} = \mu g^t + \frac{\nabla_x l(\theta, \hat{x}^t, y)}{\|\nabla_x l(\theta, \hat{x}^t, y)\|_1} \\ \hat{x}^{t+1} = \Pi_{x+S}\left(\hat{x}^t + \alpha \operatorname{sgn}(g^t).\right) \end{cases} \tag{3}$$

where g^t is the accumulated volecity vector in the gradient direction and $\mu < 1$ is the decay factor.

2.2. Defenses

Most existing methods of defense are retraining the model after adding adversarial examples to the training set [9, 11]. Some more recent works are inducted based on this idea combining with ensemble learning [21], which becomes the baseline methods of the top teams in the NIPS 2017 competition [13].

Other methods aims at the phenomenon of gradient masking [18] [17]. For example, [2] argues that using defensive distillation with different temperature T in training and test time can make gradient hard to compute. However, the attack can still be fixed by changing the gradient update as $F'(x) = \operatorname{softmax}(Z(x)/T)$.

2.3. Hypothesis and Theories

There are several interesting works on the underlying geometry of adversarial examples, which can be seen as motivations on how to design good attack/defense algorithms. An early hypothesis on the existence of adversarial examples is that the network shows a strong linearity around the input data [8]. Following this idea, [4] examines the relationship between the Lipschitz continuity of the network and its robustness, further proposes an algorithm with constraining the Lipschitz constant while training. [18] [17] discussed the gradient masking property of some trained models, which might harm the model's robustness. Recently [7] uses an artificial data to illustrate the idea of the systematic trade-off between the performance of the networks on clean data and adversarial data.

3. Method

The defense procedure can be formalized as a distributionally robust optimization problem

$$\min_{\theta\in\Theta} \sup_{P\in\mathcal{P}} E_P[l(\theta; Z)], \tag{4}$$

where $l(\theta, Z)$ is the loss function and $\mathcal{P}$ is the plausible data distribution set. In this paper, we consider the set as an Optimal Transport ball, i.e. $\mathcal{P} = \{P : W_c(P, P_0) \leq \delta\}$, where W_c is the optimal transport metric under cost function c and P_0 is the empirical distribution.

Optimal Transport metric W_c measures the closeness between two probability measures. Suppose that μ_1 and μ_2 are two probablity measures defined on a Polish space S, then the optimal transport metric under cost function $c(\cdot)$ between μ_1 and μ_2 is defined as

$$W_c(\mu_1, \mu_2) := \inf_\pi \left\{ \int c d\pi : \pi \in \Pi(\mu_1, \mu_2) \right\},$$

where $\Pi(\mu_1, \mu_2)$ denotes the family of all probability distributions defined on $S \times S$ with μ_1 and μ_2 as respective marginals. Intuitionally, the quantity $c(x, y)$ specifies the

cost of transporting unit mass from $x \in S$ to another element $y \in S$. The integral $\int cd\pi$ represents the total cost of transportation from initial distribution μ_1 to final distribution μ_2.

3.1. Strong Duality and Lagrangian relaxation

When the plausible data distribution set $\mathcal{P}$ is an optimal transport ball, the inner maximization problem $\sup_{P\in\mathcal{P}} E_P[l(\theta; Z)]$ is an infinite dimensional linear optimization problem, which is hard to optimize. Fortunately, the dual of $\sup_{P\in\mathcal{P}} E_P[l(\theta; Z)]$ can be formulated as a finite dimensional problem and the strong duality holds under mild assumptions.

Theorem 1 ([1]) *If cost function $c(x, y)$ is a nonnegative lower semicontinuous function satifying $c(x, y) = 0$ if and only if $x = y$ and $f \in L^1(dP_0)$ is upper semicontinous, the following strong duality holds*

$$\sup_{W_c(P,P_0)\le\delta} \int f dP = \inf_{\gamma\ge 0} \gamma\delta + \int \phi_\gamma dP_0,$$

where

$$\phi_\gamma(x) = \sup_y (f(y) - \gamma c(x, y)).$$

In this paper, we set $f(z) = l(\theta; z)$, so the distributionally robust optimization problem can be rewritten as

$$\min_{\theta\in\Theta} \inf_{\gamma\ge 0} \gamma\delta + E_{P_0}[\phi_\gamma(\theta; Z)], \tag{5}$$

where

$$\phi_\gamma(\theta; z) = \sup_{\hat{z}} (l(\theta; \hat{z}) - \gamma c(z, \hat{z})).$$

To make it computationally tractable for neural network, we fixed γ and consider its Lagrangian relaxation

$$\min_{\theta\in\Theta} E_{P_0}[\phi_\gamma(\theta; Z)].$$

We call this method Distributionally Robust Adversarial Training method (DRA).

3.2. Connection with Other Methods

Since cost function is free to select as long as it is lower semicontinuous, we can recover many common methods in literature using various cost functions. If the cost function is the ℓ_p-norm,

$$c((x_1, y_1), (x_2, y_2)) = \begin{cases} \|x_1 - x_2\|_p^p & y_1 = y_2 \\ +\infty & y_1 \neq y_2 \end{cases},$$

we recover [20, 2] with specific object function $f(\theta; x + r) = -l(\theta, x + r)$. If the cost function is Elastic-Net i.e.,

$$c((x_1, y_1), (x_2, y_2)) = \begin{cases} \|x_1 - x_2\|_2^2 + \beta \|x_1 - x_2\|_2 & y_1 = y_2 \\ +\infty & y_1 \neq y_2 \end{cases},$$

we recover Elastic-Net Attacks [3]. If the cost function takes the form

$$c((x_1, y_1), (x_2, y_2)) = \begin{cases} 0 & y_1 = y_2 \text{ and } \|x_1 - x_2\|_p \le \epsilon \\ +\infty & \text{else} \end{cases},$$

we recover PGM [14, 11].

3.3. Training and Defense Procedure

From the envelope theorem, we have

$$\frac{\partial \phi_\gamma(\theta; z)}{\partial \theta} = \frac{\partial l(\theta; \hat{z})}{\partial \theta},$$

where $\hat{z}$ is the solution of the inner maximum problem.

Here we adopt the algorithm from [19], which updates the neural network parameter θ dynamically during the generation process of adversarial examples.

Algorithm 1 Distributionally Robust Adversarial Training

1: **Input**: The empirical distribution P_0, learning rate α, minibatch size M, iteration times T.
2: **for** $t = 0, \ldots, T - 1$ **do**
3: Sample a minibatch from P_0 and for each image, find an maximizer ${\hat{z}_i}^t$ of $l(\theta^t; z_i) - \gamma c(z_i, z_i^t)$
4: $\theta_{t+1} \leftarrow \theta_t - \alpha\nabla \frac{1}{M} \sum_{i=1}^M l(\theta^t; {\hat{z}_i}^t))$

In practice, we iterate T_{adv} steps with learning rate α' to get the inner maximizer.

From [19], we know that this type of distributional robust adversarial training is highly sensitive to the cost function we choose. In their experiments, defense with ℓ_2 norm can successfully defend basic iterative attack [12], but defense with ℓ_∞ norm does not perform well. To address this problem, we provide the following 2 methods: multi-cost-function ensemble method and adaptive cost function method.

3.3.1 Multi-cost-function Ensemble Method

Here we consider the situation that cost function is ambiguous. Suppose we know cost function is ℓ_2 norm with probability p and cost function is ℓ_∞ norm with probability $1-p$. Then, in our training, we randomly pick the cost function with probability p in each steps to train. Here, we slightly modify the algorithm as

Algorithm 2 Distributionally Robust Adversarial Training with Multi-cost-function Ensemble

Input: The empirical distribution P_0, learning rate α, minibatch size M, iteration times T, different types of cost function $c_i(x, y)$, the probability p that cost function is 2-norm.
for $t = 0, \ldots, T-1$ **do**
  Sample a minibatch from P_0
  Generate a random variable ν that follows uniform distribution in $[0, 1]$
  if $\nu < p$ **then**
    Find an maximizer $\hat{z_i}^t$ of $l(\theta^t; z_i) - \gamma_2 \|z_i, z_i^t\|_2^2$
  else
    Find an maximizer $\hat{z_i}^t$ of $l(\theta^t; z_i) - \gamma_\infty \|z_i, z_i^t\|_\infty$
  $\theta_{t+1} \leftarrow \theta_t - \alpha \nabla \frac{1}{M} \sum_{i=1}^{M} \sum_{j=1}^{C} l(\theta^t; \hat{z_{ij}}^t))$

We expect that algorithm (2) can achieve a balance of attacks from different cost functions.

3.3.2 Adaptive Cost Function Method

For this method, we consider Mahalanobis distance, i.e.,

$$c_A((x_1, y_1), (x_2, y_2)) = \begin{cases} (x_1 - x_2)^T A (x_1 - x_2) & y_1 = y_2 \\ +\infty & y_1 \neq y_2 \end{cases}$$

where $\mathbf{1}$ is indicator function. We call A Mahalanobis matrix. And we consider a new formulation that

$$\min_{\theta \in \Theta} \max_{A \in \mathcal{A}} \mathbb{E}_{\hat{P}_n} \phi_\gamma(\theta; z),$$

where $\phi_\gamma(\theta; z) = \sup_{\hat{z}}(l(\theta; \hat{z}) - \gamma c_A(z, \hat{z}))$, and $\mathcal{A}$ is a suitable set that the adversary can choose from. Intuitively, we give adversary more power to attack under a specific direction. This method is more statistically stable and more robust to different types of attack. Here we choose

$$\mathcal{A}_r = \{A \in R^{d \times d} : A \succ 0, \sum_{i=1}^{d} \sigma_i(A)^{-r} = d\},$$

where $A \succ 0$ means A is positive semidefinite, $\sigma_i(A)$ are the eigenvalues of A, d is the dimension of data and $r > 0$. Notice that $\sum_{i=1}^{d} \sigma_i(A)^{-r} = d$ is a generalization of Frobenius norm. When $r = 2$, we recover the Frobenius norm and $\mathcal{A}_2 = \{A \in R^{d*d} : A \succ 0, \left\|A^{-1}\right\|_F = d\}$. Then, we have following theorem,

Theorem 2 *Suppose*

$$\hat{x}_i = \arg\max_u (l(\theta; (\hat{x}, \hat{y})) - \gamma c_A((x_i, y_i), (\hat{x}, \hat{y}))$$

and if we take eigendecomposition that,

$$\sum_{i=1}^{n} (\hat{x}_i - x_i)(\hat{x}_i - x_i)^T = Q^T \Psi Q,$$

then we have the optimizer A is

$$\hat{A} = \arg\max_{A \in \mathcal{A}_r} \mathbb{E}_{\hat{P}_n} \phi_\gamma(\theta; z) = Q^T \Lambda Q,$$

where

$$\Lambda_{jj} = \frac{\left(\sum_{j=1}^{d} (\Psi_{jj})^{\frac{r}{r+1}}\right)^{\frac{1}{r}}}{\Psi_{jj}^{\frac{1}{r+1}} d^{\frac{1}{r}}}.$$

Based on this theorem, we provide the following algorithm,

Algorithm 3 Adaptive Distributionally Robust Adversarial Training

Input: The empirical distribution P_0, learning rate α, minibatch size M, total epoch N, parameter for the $\mathcal{A}_r$.
Initialize A is identity matrix, i.e., $A = I_{d \times d}$
for $T = 0, \ldots, N-1$ **do**
  while iterative t until finishing one epoch **do**
    Sample a minibatch from P_0
    Find an maximizer $\hat{z_i}^t$ of $l(\theta^t; z_i) - \gamma_2 c_A(z_i, z_i^t)$
    $\theta_{t+1} \leftarrow \theta_t - \alpha \nabla \frac{1}{M} \sum_{i=1}^{M} \sum_{j=1}^{C} l(\theta^t; \hat{z_{ij}}^t))$
  Updata A according to theorem (2) at the end of each epoch.

Noticed the model does not use significantly longer time to train than Basic Distributionally Robust Adversarial Training method. Although eigendecomposition is $O(d^3)$, we only compute it once every epoch. In practice, we do not see significant change of training time.

4. Experiments and Results

Dataset We test on the CIFAR-10 data set [10]. The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. For the sake of our purpose, we do not do data augmentation or transformation, since we only focus on the robustness of the models instead of performance merely on clean dataset.

Network structure We test the defenses and attacks based on a 13-layer convolutional neural nets. It consists 10 convolutional layers and 3 fully connected layers intercepted with max pooling layers, see Figure 1 for details. This baseline model achieves 79.9% accuracy on the clean test dataset.

Evaluation metric In our study, we use the classification accuracy on untargeted white-box adversarial examples as the evaluation metric. In other words, the attacker is allowed to acquire the structure and learned parameters of the model, so that they can use gradient information to generate attack examples.

Attack methods We compare the performance of the models against the following gradient-based attacks: Fast Gradient Sign Method (1)[8], Basic Iterative Method (2) [12],

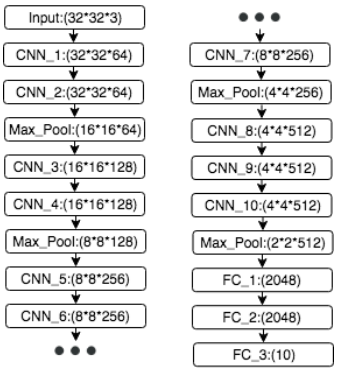

Figure 1: The baseline network architecture

Momentum Iterative Method [6] and attacks generated by our model.

Defense models For our distributionally Robust Adversarial Training method, we compare 4 methods respectively: Basic Distributionally Robust Adversarial Training with ℓ_2-norm cost function (DRA_2), Basic Distributionally Robust Adversarial Training with ℓ_∞-norm cost function(DRA_∞), Distributionally Robust Adversarial Training with Multi-cost-function Ensemble ($\text{DRA}_{2-\infty}$) and Adaptive Distributionally Robust Adversarial Training (DRA_A). We compare our models with Fast Gradient Sign Method (FGM)[8], Basic Iterative Method (BIM), Momentum Iterative Method (MIM) under the attack mentioned above.

Parameters In order to make the result comparable, we set the parameters of each attacker so that the generated adversarial images have identical distortions from the true images. The maximum attack distortion from the original image is 1.00 in ℓ_2 norm and 0.08 ℓ_∞ norm. For BIM and MIM, we choose step size 0.2 for ℓ_2 norm and 0.01 for ℓ_∞ norm; the number of attack iterations is 15 for ℓ_2 norm and 10 for ℓ_∞ norm.

The key parameters of our models are γ, which controls this distortion, and we set the γ's for $\text{DRA}_2, \text{DRA}_\infty, \text{DRA}_A$ to be 0.002, 0.02, 0.001 receptively. The γs are chosen to match the maximum attack distortion mentioned above. For $\text{DRA}_{2-\infty}$, we set $\gamma_2 = 0.002, \gamma_\infty = 0.02$ and probability $p = 0.5$ for ℓ_2 norm cost function. $r = 2$ for Mahalanobis matrix set of DRA_A which is Frobenius norm. The learning rate for inner maximum problem is 0.001 for every models and we used Adam optimizer.

We present our results based on the different choices the norms during the defense and attack. Namely, we examine the ℓ_2 norm defenses and attacks and ℓ_∞ norm defenses and attacks correspondingly. For the traditional defenses, we train the baseline model with the adversarial examples generated by different norms.

Table (1) shows our base line model attacked by various method. As we can see from the table, all the attack methods can significantly harm the baseline mode, which is learned from the vanilla training. The accuracy on the adversarial examples by DRA_2 drops down to 0.5% from 80% on the clean data.

Table 1: Base line model attacked by various method

FGM_2	FGM_∞	IGM_2	IGM_∞	MGM_2	MGM_∞	DRA_2
0.196	0.107	0.096	0.088	0.097	0.087	0.005

Table (2) gives test accuracy on clean data for various defense model. We can see all the defense strategies harm clean-data accuracy.

Table 2: Test accuracy on clean data for various defense model

Baseline	FGM_2	FGM_∞	BIM_2	BIM_∞	MIM_2	MIM_∞
0.799	0.741	0.739	0.717	0.673	0.713	0.742
	DRA_2	DRA_∞	DRA_A	$\text{DRA}_{2-\infty}$		
	0.652	0.587	0.672	0.577		

4.1. ℓ_2 Norm

The results are shown in Table (3). Each row corresponds to attacks while each column corresponds to defense models. In this subsection, we focus on ℓ_2 norm defense model under various attacks including both ℓ_2 and ℓ_∞ norm attacks.

Table 3: ℓ_2 type defense accuracy under various attacks

	FGM_2	BIM_2	MIM_2	DRA_2	DRA_A
FGM_2	0.439	0.446	0.452	0.490	0.515
FGM_∞	0.155	0.154	0.162	0.206	0.192
BIM_2	0.372	0.397	0.399	0.457	0.492
BIM_∞	0.119	0.121	0.115	0.146	0.141
MIM_2	0.384	0.406	0.407	0.462	0.497
MIM_∞	0.119	0.122	0.116	0.166	0.155
DRA_2	0.071	0.104	0.087	0.270	0.254

As for the defenses, BIM_2 and MIM_2 outperforms FGM_2, which is natural since the former 2 methods are iterative and can generate adversarial examples with stronger attacking power. Our model with DRA training beats the traditional defense in all attacks. Furthermore, our adaptive method (DRA_A) gives uniformly 2% enhancement compared to basic DRA method (DRA_2) under ℓ_2 attacks.

Row-wise speaking, the defense are more robust against the attack using the same norm when training, i.e. the performance in ℓ_2 norm attacks are consistently better than the performance in ℓ_∞ norm attacks. Further, the adversarial examples generated by the DRA_2 model have the most powerful attack on all the traditional defenses, raising lowest accuracy cross all rows.

Also, we can see that there is a trade-off between the performance in the clean data and the adversarial data across all models.

4.2. ℓ_∞ Norm

The results are shown in Table (4). Each row corresponds to attacks while each column corresponds to defense models. In this subsection, we focus on ℓ_∞ norm defense model under various attacks including both ℓ_2 and ℓ_∞ norm attacks.

Table 4: ℓ_∞ type defense accuracy under various attacks

	FGM_∞	BIM_∞	MIM_∞	DRA_∞	DRA_A	$\text{DRA}_{2-\infty}$
FGM_2	0.288	0.517	0.516	0.427	0.515	0.463
FGM_∞	0.645	0.275	0.279	0.226	0.192	0.256
IGM_2	0.139	0.496	0.497	0.407	0.492	0.439
IGM_∞	0.079	0.196	0.201	0.170	0.141	0.201
MGM_2	0.145	0.500	0.500	0.405	0.497	0.440
MGM_∞	0.062	0.196	0.204	0.184	0.155	0.212
DRA_2	0.002	0.247	0.339	0.209	0.254	0.240

The tables shows that when using ℓ_∞ norm, the most affective defend methods are the BIM_∞ and MIM_∞, which outperform our DRA_∞ framework. This is due to the fact that the infinity norm is non-differentiable in the region of interest thus usually harder to train.

However, $\text{DRA}_{2-\infty}$ indeed improve basic model DRA_∞ by 3% consistently. It means our new algorithm achieves the best in DRA with ℓ_∞ norm cost function regime. Also, DRA_A overcomes the non-differentiable difficulty, and achieves comparable performance to the best traditional models, as our newly proposed loss function can be seen as the smoothed version of the ℓ_∞ loss. Furthermore, it is easy to see that $\text{DRA}_{2-\infty}$ performs better than DRA_A under ℓ_∞ attacks, while DRA_A performs better under ℓ_2 attacks.

Quite interesting is that when using ℓ_∞ norm in the defense, the ℓ_∞ attacks still outperforms the ℓ_2 attacks in all the models except for FGM_∞. This is a strong evidence that the former attack is generally a harder problem for the defender.

5. Discussion

5.1. Visualization of adversarial examples

Some examples of the generated adversarial examples are shown in Figure (2). We can see the change is imperceptible to human eyes for ℓ_2 norm attack, while if one look carefully, he may find some tiny changes for ℓ_∞ norm attack.

(a) BIM_2 (b) DRA_2

(c) BIM_∞ (d) DRA_∞

(e) original

Figure 2: Generated adversarial examples from different attack methods.

5.2. Analysis of Mahalanobis matrix A

We showcase Mahalanobis matrix A in this subsection. The shape for A is $(32 \cdot 32 \cdot 3) \times (32 \cdot 32 \cdot 3)$ and following is part of elements of A matrix that outputs from the last iteration. We can see A matrix indeed capture some correlation structure. If $i - j$ is small, then A_{ij} has a

large negative value. The structure of A stimulates adversary to attack the adjacent nodes along the same direction.

$$\begin{bmatrix} 1.865 & -0.731 & -0.378 & ... & 0.001 & 0. & 0.005 \\ -0.731 & 2.33 & -0.924 & ... & -0.006 & 0.009 & -0.006 \\ -0.378 & -0.924 & 2.993 & ... & 0.005 & -0.005 & -0.004 \\ ... & & & & & & \\ 0.001 & -0.006 & 0.005 & ... & 3.031 & -1.041 & -0.462 \\ 0. & 0.009 & -0.005 & ... & -1.041 & 2.758 & -1.062 \\ 0.005 & -0.006 & -0.004 & ... & -0.462 & -1.062 & 2.776 \end{bmatrix}$$

6. Conclusion and Future Work

In this paper, we compare DRA type model with other heuristic methods and propose a new defense scheme DRA_A which is more statistically robust and stable. For ℓ_2 attacks, DRA_A achieve the state-of-art results.

However, we did not fully understand the structure of Mahalanobis marix A. We can investigate the information carried by A to better understand adversarial attack and defense in future work. Another thing is that DRA type model with ℓ_∞ cost function does not perform well because it is hard to optimize. We will resort to more powerful optimizer to handle this type of problems.

7. Appendix: Proof of Theorem (2)

First we give a lemma.

Lemma 3 *For Ψ is positive definite diagonal matrix and $r > 0$,*

$$\min_{Q\in O(d)} \left(\sum_{j=1}^{d} \left((Q^T \Psi Q)_{jj} \right)^{\frac{r}{r+1}} \right) = \sum_{j=1}^{d} (\Psi_{jj})^{\frac{r}{r+1}}$$

where $O(d)$ is the space of $d \times d$ orthogonal matrix.

Proof:

$$\sum_{j=1}^{d} \left(Q^T \Psi Q\right)^{\frac{r}{r+1}} = \sum_{j=1}^{d} \left(q_j^T \Psi q_j\right)^{\frac{r}{r+1}} = \sum_{j=1}^{d} \left(\sum_{i=1}^{d} \Psi_{ii} q_{ji}^2 \right)^{\frac{r}{r+1}}$$

where $Q = [q_1, q_2, \ldots, q_d]$. Since $\phi(x) = x^{\frac{r}{r+1}}$ is concave function. according to Jensen inequality, we have

$$\left(\sum_{i=1}^{d} \Psi_{ii} q_{ji}^2 \right)^{\frac{r}{r+1}} \geq \sum_{i=1}^{d} q_{ji}^2 \Psi_{ii}^{\frac{r}{r+1}},$$

then,

$$\sum_{j=1}^{d} \left(\sum_{i=1}^{d} \Psi_{ii} q_{ji}^2 \right)^{\frac{r}{r+1}} \geq \sum_{j=1}^{d} \sum_{i=1}^{d} q_{ji}^2 \Psi_{ii}^{\frac{r}{r+1}} = \sum_{i=1}^{d} \Psi_{ii}^{\frac{r}{r+1}}.$$

□

Based on the lemma, we give the proof.

Proof of theorem (2):

$$\hat{A} = \arg\max_{A\in\mathcal{A}} \mathbb{E}_{\hat{P}_n} \phi_\gamma(\theta, z) = \arg\min_{A\in\mathcal{A}} \sum_{i=1}^{n} (\hat{u}_i - x_i)^T A (\hat{u}_i - x_i)$$

By denoting $\zeta_i = \hat{u}_i - x_i$, we refomulate the problem as $\min_{A\in\mathcal{A}_r^d} \sum_{i=1}^{n} \zeta_i^T A \zeta_i$. The eigendecomposition is that $A = Q^T \Lambda Q$, and denote $\tilde{\zeta}_i = Q\zeta_i$. Then, after further simplification, the problem becomes

$$\begin{aligned} & \min_{Q\in O(d)} \min_{\substack{\Lambda = diag(a_1, a_2, \ldots a_d) \\ \sum_{i=1}^{d} a_i^{-r} = d}} \sum_{i=1}^{n} \tilde{\zeta}_i \Lambda \tilde{\zeta}_i \\ = & \min_{Q\in O(d)} \min_{\sum_{i=1}^{d} a_i^{-r} = d} \sum_{j=1}^{d} \left(\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 \right) a_j, \end{aligned}$$

where $O(d)$ is the space of $d \times d$ orthogonal matrix. We first deal with the inner minimization problem,

$$\min \sum_{j=1}^{d} \left(\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 \right) a_j \; s.t. \sum_{j=1}^{d} a_j^{-r} = d.$$

By using Hölder inequality, we have

$$\begin{aligned} & \left(\sum_{j=1}^{d} \left(\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 \right) a_j \right)^{\frac{r}{r+1}} \\ \geq & \left(\sum_{j=1}^{d} \left(\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 \right)^{\frac{r}{r+1}} \right) / \left(\sum_{j=1}^{d} \frac{1}{a_j^r} \right)^{\frac{1}{r+1}} \\ = & d^{-1/r} \left(\sum_{j=1}^{d} \left(\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 \right)^{\frac{r}{r+1}} \right) \end{aligned}$$

And equality holds when

$$\frac{\left(\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 \right) a_j}{\frac{1}{a_j^r}} = \left(\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 \right) a_j^{r+1} = t \Leftrightarrow a_j = \sqrt[r+1]{\frac{t}{\left(\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 \right)}}$$

Then for outer minimization problem, we have

$$\min_{Q\in O(d)} d^{-\frac{1}{r}} \left(\sum_{j=1}^{d} \left(\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 \right)^{\frac{r}{r+1}} \right)^{\frac{r+1}{r}},$$

Suppose $\boldsymbol{\zeta} = [\zeta_1, \zeta_2, \ldots, \zeta_n]$ and

$$\sum_{i=1}^{n} (\hat{u}_i - x_i)(\hat{u}_i - x_i)^T = \boldsymbol{\zeta}\boldsymbol{\zeta}^T = S^T \Psi S.$$

We have

$$\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 = \left(Q\zeta\zeta^T Q^T\right)_{jj} = \left(QS^T \Psi S Q^T\right)_{jj}$$

and denote $\hat{Q} = QS^T$.Therefore, the above problem is equivalent to

$$\min_{\hat{Q}\in O(d)} \left(\sum_{j=1}^{d} \left(\left(\hat{Q}^T \Psi \hat{Q}\right)_{jj}\right)^{\frac{r}{r+1}}\right)^{\frac{r+1}{r}}$$

Then, According lemma 3, we have

$$\min_{\hat{Q}\in O(d)} \left(\sum_{j=1}^{d} \left(\left(\hat{Q}^T \Psi \hat{Q}\right)_{jj}\right)^{\frac{r}{r+1}}\right)^{\frac{r+1}{r}} = \left(\sum_{j=1}^{d} \left(\left(\Psi\right)_{jj}\right)^{\frac{r}{r+1}}\right)^{\frac{r+1}{r}}$$

$$\arg\min_{\hat{Q}\in O(d)} \sum_{j=1}^{d} \left(\left(\hat{Q}^T \Psi \hat{Q}\right)_{jj}\right)^{\frac{r}{r+1}} = I_d.$$

Therefore,

$$Q = \left(S^T\right)^{-1} = S.$$

Furthermore,

$$\sum_{i=1}^{n} \tilde{\zeta}_{ij}^2 = \Psi_{jj} \Leftrightarrow a_j = \sqrt[r+1]{\frac{t}{\Psi_{jj}}},$$

Since we have

$$\sum_{j=1}^{d} a_j^{-r} = d \Leftrightarrow \sum_{j=1}^{d} \left(\frac{\Psi_{jj}}{t}\right)^{\frac{r}{r+1}} = d.$$

and then

$$t = \left(\frac{\sum_{j=1}^{d} \left(\Psi_{jj}\right)^{\frac{r}{r+1}}}{d}\right)^{\frac{r+1}{r}}, a_j = \frac{\left(\sum_{j=1}^{d} \left(\Psi_{jj}\right)^{\frac{r}{r+1}}\right)^{\frac{1}{r}}}{\Psi_{jj}^{\frac{1}{r+1}} d^{\frac{1}{r}}}.$$

□

Acknowledgments We would like to express our appreciations to Jose Blanchet and Hongseok Namkoong for many useful inputs and valuable comments. We used the code of

`https://github.com/tensorflow/cleverhans`

`https://github.com/ricvolpi/certified-distributional-robustness`

and thank the authors. We also thank the instructor and TA teams of CS231N for their clearly delivery of lectures and many helps and Google Cloud for their generosity to provide GPU.

Contributions Nian proposed the Adaptive Cost Function Method, implemented the method and did the experiments. Fan and Teng did the experiments, wrote the paper and provided many useful insights.

References

[1] J. Blanchet and K. R. Murthy. Quantifying distributional model risk via optimal transport. *arXiv preprint arXiv:1604.01446*, 2016.

[2] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017.

[3] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh. Ead: elastic-net attacks to deep neural networks via adversarial examples. *arXiv preprint arXiv:1709.04114*, 2017.

[4] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning*, pages 854–863, 2017.

[5] Y. Dong, F. Liao, T. Pang, X. Hu, and J. Zhu. Discovering , adversarial examples with momentum. *arXiv preprint arXiv:1710.06081*, 2017.

[6] Y. Dong, F. Liao, T. Pang, H. Su, X. Hu, J. Li, and J. Zhu. Boosting adversarial attacks with momentum.

[7] J. Gilmer, L. Metz, F. Faghri, S. S. Schoenholz, M. Raghu, M. Wattenberg, and I. Goodfellow. Adversarial spheres. *arXiv preprint arXiv:1801.02774*, 2018.

[8] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. pages 1–11, 2014.

[10] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009.

[11] A. Kurakin, G. Brain, I. J. Goodfellow, and S. Bengio. Adversarial Machine Learning At Scale. pages 1–17, 2017.

[12] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[13] A. Kurakin, I. Goodfellow, S. Bengio, Y. Dong, F. Liao, M. Liang, T. Pang, J. Zhu, X. Hu, C. Xie, et al. Adversarial attacks and defences competition. *arXiv preprint arXiv:1804.00097*, 2018.

[14] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. pages 1–27, 2017.

[15] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. DeepFool: a simple and accurate method to fool deep neural networks. 2015.

[16] S. M. Moosavi Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, number EPFL-CONF-218057, 2016.

[17] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM, 2017.

[18] N. Papernot, P. McDaniel, A. Sinha, and M. Wellman. Towards the science of security and privacy in machine learning. *arXiv preprint arXiv:1611.03814*, 2016.

[19] A. Sinha, H. Namkoong, and J. Duchi. Certifiable distributional robustness with principled adversarial training. *arXiv*

preprint arXiv:1710.10571, 2017.

[20] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[21] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.